بسم الله الرحمن الرحيم

سلسلة تعلم ORACLE بسهولة

الدرسُ الأول — مقدمة لأوراكل

تمهيد :

نستنتج من الدروس السابقة عن مفاهيم علم قواعد البيانات ؛ أنَّ قواعد البيانات عبارة عن تجميع لكمية كبيرة من المعلومات أوالبيانات وعرضها بطريقة أو أكثر من طريقة ليسهل الاستفادة منها .

وتشترك معظم نظم إدارة قواعد البيانات في مجموعة من الوظائف منها :

أ- إضافة معلومة أو بيانات إلى الملف .

ب- حذف البيانات القديمة .

جـ - تغيير البيانات الموجودة .

د- ترتيب وتنظيم البيانات داخل الملفات .

هـ - عرض البيانات على شكل تقرير أو نموذج .

وكتطبيق لهذه المفاهيم ، سنتعلم لغة أوراكل من الصفر بإذن الله تعالى ..

إنَّ الاوراكل لاتعتبر لغة برمجة وإنما هي لغة قواعد بيانات مبرمجة لقواعد البيانات فيجب علينا ان نعرف هذا الفرق الاساسي بين لغة قواعد البيانات الاوراكل وبين لغات البرمجة الاخرى، فهي شبيهة إلى Microsoft Access وأقرب إلى Microsoft SQL Server ، يعني أنها قاعدة بيانات وليست لغة برمجة مستقلة بحد ذاتها فمثلا فجوال بيسك تعتبر لغة برمجة لأنها تمكنك من عمل برامج ذات أهداف متعددة ولا يشترط أن تكون برامج قواعد بيانات فيمكنك عمل برنامج رسم كما برنامج الرسام في ويندوز ويمكنك التحكم بها على حد كبير من المرونة والإمكانيات التي تسمح لك حتى بالارتباط بقواعد بيانات متعددة ولكن أوراكل تتميز ببميزات عالية تميزها عن غيرها من لغات قواعد البيانات نذكر منها :

- أنها قاعدة بيانات قوية و آمنة ؛ إذ تتمتع بأمان عالي جداً ، وهو سبب أساسي لانتشارها الهائل رغم التكلفة الباهظة لها..

- انها تعتبر قواعد بيانات ضخمة ، مقارنة بغيرها .
- يوجد لديها أدوات تساعدها للتعامل معها وإظهارها في أشكال متعددة ، بما يسمى تطبيقات أوراكل ؛ أي الـ DEVELOPER ، حيث يمكنك إدخال البيانات واستخراجها عن طريق نماذج وتقارير ورسوم بيانية ولكن لا يمكنها التعامل مع قاعدة بيانات غير أوراكل كما أنها لا تمكنك من عمل برامج مثل الرسام .

و يوجد نفرعين لمن اراد تعلم الاوراكل كالتالي :

١- مدير قواعد البيانات الاوراكل (Administer) .

٢- مطور قواعد البيانات الاوراكل (Developer) .

ونحنُ بصدد التعلم للوصول إلى مطور قواعد البيانات ، وحتى نتعلمه، يجب أن نتقن لغة SQL و PL/SQL ..ثم ندخل في الـ DEVELOPER بجزأيه الـ FORM والـ REPORT.. وسوف نبدأ هنا بكورس متواضع عن الـ SQL ثم PL/SQL ..

لغة الإستعلام البنيوية أو الهيكلية

Structured Query Language

عبارة عن مجموعة من الأوامر التي تحتاجها البرامج وكذلك المستخدمين للوصول للبيانات الموجودة ضمن قاعدة بيانات أوراكل ..

تم تطوير هذه اللغة البداية من قبل شركة IBM وذلك في منتصف السبعينات ، وكانت تسمى System R حيث كانت عبارة عن نموذج لنظام إدارة قواعد بيانات علاقئية .

بعدها تم توصيف لغة SQL في ١٩٧٦م في مجلة INM Journal of R&D باسم SEQUEL2 ومن ثمَّ قامت شركة ORACLE في ١٩٧٩م إنزال أول نسخة تجارية من لغة SQL .

ولغة SQL هي عبارة عن لغة غير إجرائية Non-Procedural Language ،لأنها تتعامل مع مجموعة سجلات في الوقت نفسه وليس مع سجل وحيد كما أنها تمكننا من استكشاف البيانات تلقائيا .

وتحتوي لغة SQL على تعليمات تفيد المستخدمين وتمكنهم من إدارة النظام وقواعد البيانات والتطبيقات عليها ، كما أنها تحتوي على أوامر لإنجاز مهام مختلفة ومتعددة كالبحث عن البيانات والتعامل مع قواعد البيانات والسجلات وتضمن تناسق وتكاملية الببيانات ..

ولكن مالفرق بين SQL و SQL*Plus ؟

وكما نعلم أن SQL عبارة عن لغة تعليمات للإتصال بمخدم أوراكل Oracle Server من خلال أية أداة أو أي تطبيق ، وعندما تقوم بكتابة تعليمة SQL ، يتم تخزينها في جزء من الذاكرة يسمى ذاكرة SQL المؤقتة SQL buffer وتبقى فيها حتى تقوم بكتابة تعليمة جديدة .

أما SQL*Plus فهي عبارة عن أداة من أدوات أوراكل يمكنها التعرف على تعليمات SQL وإرسالها إلى مخدم أوراكل لتنفيذها ،وهي تمتلك تعليمات إضافية خاصة بها سنقوم بشرحها في الدرس الثاني إن شاء الله .

وهذه اللغة نتعامل من خلالها مع قاعدة البيانات اوراكل ، اي انه من خلال هذه اللغة نستطيع إعطاء الصلاحيات لمستخدم بإعطائه صلاحية الاتصال بقاعدة البيانات (GRANT) ، ومنحه صلاحيات وامتيازات ممارسة عمليات معينة (REVOKE) ، كذلك بإمكاننا إنشاء الجداول (CREATE) ، والتعديل عليها (ALTER) ، وحذف الجداول (DROP) الغير مرغوب فيها ، وكذلك نستطيع ملء الجداول بالبيانات (INSERT) ، والتعديل على البيانات المدخلة (UPDATE) ، وحذف أي بيانات (DELETE) غير مرغوب فيها. وبعد أن تكتمل لدينا الجداول نستطيع الاستعلام عن البيانات المدخلة وذلك بالأمر (SELECT) .

مما سبق يمكننا تقسيم أوامر SQL إلى ثلاثة أقسام ،حيثُ تشكل كل مجموعة أوامر لغة فرعية من هذه اللغة وهي كالتالي :

١- أوامر لغة تعريف البيانات Data Definition Language (DDL) :-

وتحتوي على ثلاث أوامر وهي :

- يستخدم لإنشاء الجداول (CREATE TABLE).
- يستخدم للتعديل على جدول منشأ سابقاً (ALTER TABLE).
- يستخدم لحذف جدول غير مرغوب فيه (DROP TABLE).

حيث يقتصر عمل هذه الأوامر على الجداول وحقولها فقط دون التعرض للبيانات التي بداخل الجداول .

٢- أوامر لغة معالجة البيانات Data Manipulation Language (DML) :-

وتحتوي على أربع أوامر وهي :

- أمر إدخال البيانات إلى الجدول (INSERT INTO).
- أمر التعديل على بيانات الجدول (UPDATE) .
- أمر حذف بيانات من الجدول (DELETE) .
- أمر الاستعلام عن شيء معين في بيانات الجدول (SELECT) .

وتعمل هذه الأوامر على البيانات التي بداخل الجداول وذلك من إدخال بيانات ، وتعديل بيانات مدخله ، وحذف بيانات مدخله والاستعلام عن بيانات معينه .

٣- أوامر لغة التحكم بالبيانات Data Control Language (DCL) :-

وتحتوي على أمرين اثنين هما :

- أمر إعطاء الامتيازات والصلاحيات (GRANT) .
- أمر منح الامتيازات والصلاحيات (REVOKE) .

وسنقوم بدراسة كل أمر على حده والتعرف على وظيفته وكيفية التعامل معه وكتابته في دروسنا القادمة بإذن الله تعالى ..

ملخص الدرس :

- تتميز أوراكل ببميزات عالية تميزها عن غيرها من لغات قواعد البيانات .
- لغة SQL هي عبارة عن لغة غير إجرائية (Non-Procedural Language) ،لأنها تتعامل مع مجموعة سجلات في الوقت نفسه وليس مع سجل وحيد ، وهي لغة تعليمات للإتصال بمخدم أوراكل Oracle Server من خلال أية أداة أو أي تطبيق ..
- SQL*Plus عبارة عن أداة من أدوات أوراكل ؛ يمكنها التعرف على تعليمات SQL وإرسالها إلى مخدم أوراكل لتنفيذها ،وهي تمتلك تعليمات إضافية خاصة بها.
- تُقسم أوامر SQL إلى ثلاثة أقسام :

١- أوامر تعريف البيانات (DDL) .

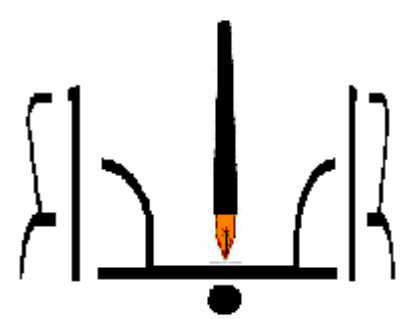
٢- أوامر معالجة البيانات (DML) .

٣- أوامر التحكم بالبيانات (DCL).

وفي الدرس القادم بإذن الله سنتحدث عن كيفية الدخول إلى SQL * Plus 3.3 وخطوات إنشاء مستخدم جديد ، فإلى لقاء قريب بإذن الله تعالى .

وسامحونا على التقصير ، وفق الله الجميع لما يحبه ويرضاه ..

الأسيـــــــــــــــف



Email:amaar1422@hotmail.com

بسم الله الرحمن الرحيم

سلسلة تعلم ORACLE بسهولة

الدرسُ الثاني

بيئة SQL * Plus 3.3 وكيفية إنشاء مستخدم جديد

عادةً إن لم يكن دائماً ما تبدأ دروس SQL التي نقرأها ؛ بشرح أوامر لغة معالجة البيانات Data Manipulation Language (DML) ، وبالأخص تعليمة الإستعلام عن البيانات و استعراضها Select ..لكننا هنا في هذه السلسة المتواضعة ، نود أن نخطوا على منهاج أوراكل بقدر المستطاع ، فنبدأ بشرح قوائم اللغة نفسها وأوامر التحرير والملفات فيها ، حتى يكون الأساس لدى المتعلم قوياً ومتوافقاً مع منهجية أوراكل ..

لذلك سنشرح هذا الأساس على درسين ، أولاهما نتحدث فيه عن قوائم SQL * Plus 3.3 ، وكيفية إنشاء مستخدم خاص باسمنا ، ومنحه صلاحية الاتصال بقاعدة البيانات ..وثانيها عن أوامر القوائم التحرير والملفات في SQL * Plus 3.3 ..ونبدأ على بركة الله ..

* كيفية الدخول إلى SQL * Plus 3.3 :-

عند تحميلك للغة ، ستجد أيقونة في سطح المكتب باسم SQL * Plus 3.3 ...

وعند تشغيلك لها تظهر لك شاشة تقوم بسؤالك عن اسم المستخدم User name وكلمة المرور Password وتعريف اسم قاعدة البيانات Database في حال كنت تستخدم النسخة الخاصة بالكمبيوتر الشخصي فيمكنك استخدام اسم المستخدم وكلمة المرور فقط في حال لم يكن لديك أكثر من قاعدة بيانات الوضع الافتراضي عند تشغيله لأول مرة هو كتابة اسم المستخدم وكلمة المرور ولكن في حال كمبيوتر الشبكة فمن المستحسن كتابة اسم قاعدة البيانات..ويمكنك اختيار أحد هذه الأسماء للعمل بها حتى تُنشئ مستخدم خاص بك ..كمايلي :

* الاسم الاول system ؛ وهو ما يمثل أواركل أو النظام بمعنى أصح ..وهو افتراضي ..

اسم المستخدم system
كلمة المرور manager

أو من خلال الاسم الثاني وهو المستخدم الافتراضي scott ..

اسم المستخدم scott
كلمة المرور tiger

وكلمة السر هذه هي التي تكون بها مديراً لقاعدة البيانات والتي يمكنك عن طريقه الحصول على كافة السماحيات ويمكنك من تعريف حساب مستخدم جديد خاص بك ..كما تستخدم تستخدم لإعطائك بعض السماحيات (الصلاحيات) مثل الاتصال والإضافة والحذف والتعديل والاستعلام ..والآن سنتلم كيفية إنشاء مستخدم خاص بك ..

* خطوات إنشاء مستخدم جديد ومنحه بعض الصلاحيات والامتيازات :

عرفنا سابقاً أن لغة DCL هي المسؤولة عن ذلك باستخدام الامرين grant و revoke ،وحتى نستطيع انشاء مستخدم جديد ، سنحتاج حالياً للأمر grant ، وفي دروسٍ متقدمة إن شاء الله ؛ سنتتخدم الأمر revoke لإعطاء صلاحيات العمليات له ..وفيما يلي خطوات إنشاء مستخدم جديد ومنحه بعض الصلاحيات والامتيازات ، لكن لن تتقن هذه الخطوات أخي الفاضل ؛ حتى تطبقها حرفياً ، وخطوة خطوة مع بعضنا البعض ..

١- نبدأ بكتابة الأمر التالي .. وليكن المستخدم الذي نريد انشاءه هو : ali بكلمة السر aaa ..(طبعاً كل واحد يُنشيء مستخدم باسمه ، وكلمة سر سهلة ، حتى لا ينساها) ..

```
SQL> create user ali identified by aaa ;
```

ومعناه : أنشئ مستخدم بالإسم ali ، يكون معرفاً بكلمة السر : aaa ..
ولكن ستظهر لك رسالة الخطأ التالية ..

```
ERROR at line 1 :
ORA-01031 :صلاحيات غير كافية
```

ومفاد هذه الرسالة : أنك ليست لديك أي صلاحية حتى تُنشئ مستخدم جديد ، والسبب هو عدم اتصالك بأوراكل ، أو بمعنى أصح بالنظام ..فيجبُ الاتصال بالنظام ..

٢- يكون الاتصال بالنظام system من خلال الأمر ..

```
SQL> connect system
```

وكذلك عندما تريد أن تتصل بأي مستخدم..

```
SQL> connect اسم المستخدم
```

بدون فاصلة منقوطة ، وبمجرد الضغط على مفتاح الادخال (Enter) يظهر لنا السطر التالي ، طالباً منا ادخال كلمة السر (password) ..

Enter password : هنا ندخل كلمة السر

فندخل له كلمة السر الخاصة بـ system وهي manager كمايلي ..

Enter password : *******

فتظهر لنا العبارة التالية ..

Connected .

والتي تدل على أننا اتصلنا بالنظام ، ويمكننا إجراء مانريده من عمليات ..

٣- نعيد كتابة أمر إنشاء مستخدم جديد ..

SQL> create user ali identified by aaa ;

فتظهر لنا العبارة التالية والتي تخبرنا أنه تم إنشاء مستخدم ..

User created.

والآن نعطي المستخدم ali صلاحية الاتصال بالنظام ، من خلال الأمر grant ..

SQL> grant connect to ali;

ومعناها اعطي المستخدم ali حق الاتصال بالنظام ..

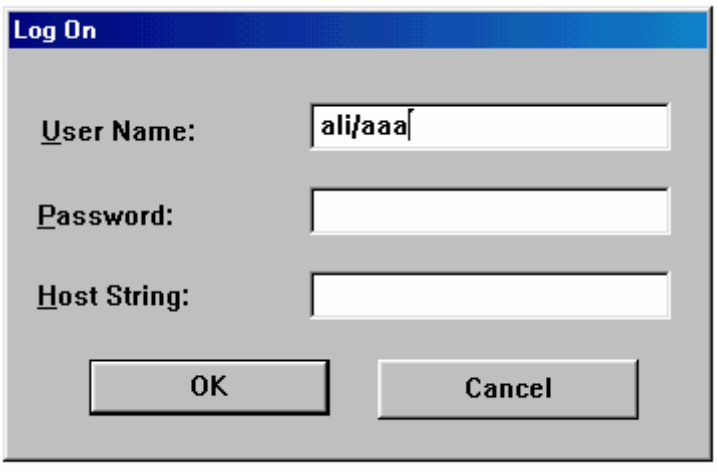
فتظهر لنا الرسالة التالية ..

Grant succeeded.

وهي تخبرنا ، أنه تم اعطاء المستخدم ali حق الاتصال بالنظام ..

وبذلك نكون استطعنا إنشاء مستخدم ، بكلمة سرٍ معينة ، وإعطائه حق الإتصال بالنظام ، وبقي أن تجرب هذه الصلاحية ..كمايلي ..

قم بالخروج نهائياً من SQL ، ثم ادخل مرة أخرى ، وعند ظهورشاشة الدخول قم بادخال اسم المستخدم الذي أنشأته متبوعا بـ / ثم كلمة السر ، أو تكتب كلمة السر في الخانة الثانية ..كلها صحيحة ..ثم اضغط على زر OK ..عند ذلك تدخل إلى النظام ..



في حالة ما إذا كان اسم المستخدم خاطئاً ، أو كانت كلمة السر خاطئة ستدخل إلى بيئة SQL * Plus 3.3 ، ولكن سُتطالب بإدخال اسم المستخدم وكلمة السر ..

ملخص الدرس :

- عند الدخول إلى (SQL * Plus 3.3) ندخل اسم المستخدم ، وكلمة السر ، مجموعة العمل في الشبكة إن كانت موجودة ..
- يوجد في بيئة (SQL * Plus 3.3)مستخدمين افترافضين هما : system بكلمة السر manager : و scott بكلمة السر : tiger..
- صيغة إنشاء مستخدم في بيئة (SQL * Plus 3.3) هي :

```
SQL> create user name user identified by password ;
```

حيثُ name user اسم للمستخدم ، و password كلمة السر الخاصة به .

- لايُسمح بإنشاء مستخدم ومنحه صلاحيات إلا بعد الاتصال بالنظام ، ويكونُ الاتصال بالنظام و بأي مستخدم بالصيغة التالية ..

```
SQL> connect اسم المستخدم
```

- لإعطاء صلاحية الاتصال لمستخدم ، نستخدم الصيغة التالية .

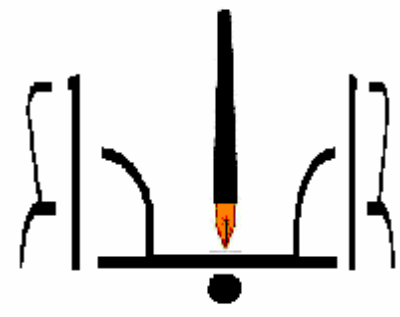
```
SQL> grant connect to name user ;
```

حيثُ name user اسم للمستخدم ..

****

وفي الدرس القادم بإذن اللّه سنتحدث عن أوامر القوائم و التحرير والملفات في بيئة SQL * Plus 3.3 ، فإلى لقاء قريب بإذن اللّه تعالى .وسامحونا على التقصير ، وفق اللّه الجميع لما يحبه ويرضاه ..

الأسيـــــــــــــــف



Email:amaar1422@hotmail.com

بسم الله الرحمن الرحيم

سلسلة تعلم ORACLE بسهولة

الدرسُ الثالث — أوامر التحرير في بيئة SQL * Plus 3.3

سنشرحُ في هذا الدرس بإذن اللّه ؛ أوامر التحرير والملفات في بيئة SQL * Plus 3.3 ..

أولاً : أوامر التحرير :-

يمكنك بعد كتابة أوامر SQL ، إجراء عمليات التحرير عليها باستخدام أوامر SQL * Plus 3.3 ، وسنقوم الآن بشرح هذه الأوامر مع إعطاء الأمثلة الموضحة :

- **أمر الإضافة** Append :

إن الشكل العام للتعليمة هو ..

sql>A **ماسنضيفه**

فمثلاً : لدينا التعليمة التالية : sql>select ename from

وأردنا كتابة اسم الجدول emp بعد from نكتب :

Sql>A emp ;

وهنا تصبح التعليمة على الشكل :

sql>select ename from emp ;

- **أمر التعديل** Change :

يسمح هذا الأمر بتغيير كلمةٍ ما ضمن التعليمة بكلمة أخرى ، ونستفيد من هذا الأمر عند كتابة اسم جدول أو حقل بالخطأ ، وذلك لأننا في بيئة SQL * Plus 3.3 لانستطيع الرجوع إلى السطر السابق في حالة كتابة التعليمات ..

إن الشكل العام للتعليمة هو ..

```
sql> C  / old   /new
```

وهذا هو الشكل القياسي لها ..حيثُ old الكلمة المُراد استبدالها ، و new الكلمة الجديدة .

فإذا كانت لدينا العبارة التاية :

```
sql>select enamee from  emp dept
```

وأردنا تغيير كلمة enamee إلى ename نكتبُ الأمر :

```
sql> c / enamee / ename
```

وهناك صيغة ثانية ، عندما نريد حذف نص ما ، وذلك بكتابة الأمر :

```
sql>C[HANGE] /  text  /
```

مثلاً لتكن لدينا مثلاً العبارة التالية :

```
sql>select ename from  emp  dept ;
```

فإذا أردنا حذف كلمة dept نكتب :

```
sql> c / dept /
```

* لاحظ أننالم نكتب شيء بعد الـ / الثانية لا ستبدالها بالمحذوف لأننا نريد الحذف فقط ،

فتصبح العبارة بعد الحذف :

```
sql>select ename from  emp  ;
```

- **أمر مسح دارئ [مخزن مؤقت]** Clear buffer SQL :

كما ذكرنا في درسنا السابق فإنه يتم تخزين أي تعليمة من تعليمات SQL في ذاكرة مؤقتة تسمى دارئ SQL buffer ، ويكمننا مسحُ هذه الذاركة بكتابة الأمر :

```
sql>clear buffer (cl buff);
```

واختصاراً للأمر نكتب مابداخل القوس فقط ، فمثلاً لتكن العبارة التالية :

```
sql>select * from tab ;
```

هذا الأمر يقوم باستعراض أسماء جميع الجدوال الموجودة في قاعدة البيانات ..وناتج هذا الأمر ، الذي هو أسماء الجدوال ، سيظهر على الشاة ويُخزن مؤقتاً في SQL buffer ..فإذا أردنا مسح هذه القيمة المخزنة نكتب ..

```
SQL> cl buff ;
```

فتظهر لنا العبارة التاية والتي تخبرنا بمسح هذه الذاكرة :

```
buffer cleared
```

E ويمكن اختصار هذا الأمر بالضغط على مفتاحي shift +Delete ..فتظهر لنا الرسالة التالية ..

****

ثانياً : أوامر الملفات :-

- **أمر الحفظ** Save :

يمكن حفظ محتوى ذاكرة SQL المؤقتة في ملف بكتابة الأمر ..

```
sql>sav filename.sql
```

E حيثُ filename.sql اسم الملف المراد الحفظ به ، ويلزم بالإمتداد sql ..

فمثلاً : لو استعرضنا جميع الجداول الموجودة في قاعدة البيانات ، ولكن للمستخدم SCOOT بكلمة السر الخاصة بها tiger ..سنجد النتيجة التالية ..

```
SQL> select * from tab ;

TNAME                          TABTYPE CLUSTERID
-------- ------- ------------------------------
BONUS                          TABLE
DEPT                           TABLE
EMP                            TABLE
SALGRADE                       TABLE
SECTION                        TABLE
STUDENT                        TABLE

6 rows selected ;
```

وأردنا حفظ هذه تعليمة Select وما ستجلبه من نتائج ..نكتب الأمر التالي :

```
sql>sav ali.sql ;
```

فستظهر لنا العبارة التالية ..

```
Created file ali.sql
```

والتي تخبرنا أنه تم حفظ التعليمة بملف اسمه ali.sql ...

وسنتأكد من نجاح هذه التعليمة ، عندما نجلب الملف الذي حفظناه في التعليمة التالية ..

- **أمر جلب محتوى ملف** Get :

يستدعي هذا الأمر محتوى آخر ملف تم حفظه ضمن ذاكرة SQL المؤقتة ..والشكل العام لهذا الأمر ..

```
sql>get  filename.sql
```

حيثُ filename.sql اسم آخر ملف تم حفظه ضمن ذاكرة SQL المؤقتة ..

ولنقم الآن بجلب الملف الذي حفظناه سابقاً بالاسم ali.sql ..

```
SQL> get ali.sql;
```

والنتيجة هي بيان ناتج آخر استعلام ..

```
1* select * from tab
```

وبمجرد تنفيذ التعليمة بكتابة / بعد محث SQL ..

```
SQL> /
```

فيظهر لنا نتيجة الإستعلام ، كما رأيناها قبل خزن الملف ..

```
TNAME                          TABTYPE CLUSTERID
------------------------------ ------- ---------
BONUS                          TABLE
DEPT                           TABLE
EMP                            TABLE
SALGRADE                       TABLE
SECTION                        TABLE
STUDENT                        TABLE

6 rows selected.
```

- **أمر تنفيذ محتوى ملف** Start :

يمكننا القيام بتنفيذ محتوى ملف ؛ تم حفظه من قبل بكتابة الأمر :

```
sql>star  filename.sql
```

او الامر ..

```
sql> @ filename.sql
```

فمثلاً إذا أردنا تنفيذ محتوى الملف ali.sql نكتب الأمر ..

```
SQL> star ali.sql;
```

أو الأمر ..

```
SQL> @ ali.sql;
```

وستكون لدينا نفس النتيجة طبعاً ..

- **أمر تشغيل برنامج التحرير Edit :**

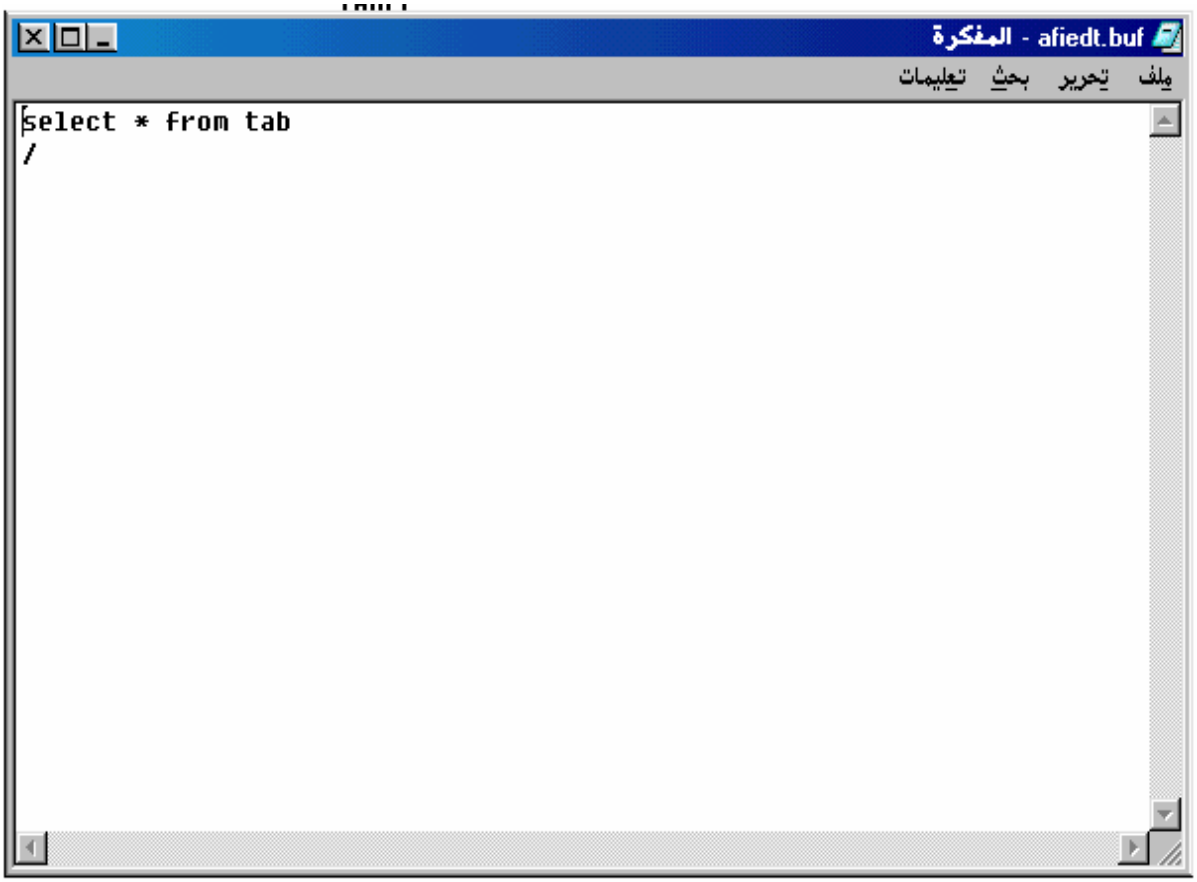
هذا الأمر هام ويكثر استخدامه ، فبواسطته نستطيع تحرير وتعديل محتوى تعليمات ملف ما.
فإذا كان الملف الحالي نكتب :

```
SQL> ed
```

ثم نضغط مفتاح Enter ..
فستظهر لنا العبارة التالية ..

```
Wrote file afiedt.buf
```

والتي تدل على الملف الذي كُتب ، بفتح نافذة المفكرة (برنامج التحرير الخاص بالـWindows) :



ويظهر مكتوباً داخلها ؛ التعليمة التي تم كتابتها في الملف ..وعندها نستطيع التعديل والإضافة والحذف في التعليمات ، لأننا لا نستطيع مسح تعليمة كتبناها في بيئة SQL بعد ان تجاوزناها بسطر أو أكثر ، ويفضل البعض كتابة تعليمات SQL في المفكرة ثم نسخها ولصقها

في بيئة SQL ، وهذا تفضيل خاطئ بل هو ناجم عن الكسل في الكتابة اللحظية لكل تعليمة وتنفيذها في وقتها ..

E ولكن لماذا هذا التفضيل خاطئ ، لأسبابٍ منها :

- حتى تتعلم أخطائك حين الكتابة ..
- تتعود على كتابة التعليمات .
- وتتعرف على رسائل الخطأ وكيفية التعامل معها ..

أما لتحرير ملفٍ ما ، ومحفوظ باسم معين .. نكتب ..

```
SQL> ed ali.sql;
```

وبنفس الخطوات السابقة ..

وبعد انتهائنا من التحرير في بيئة المفكرة ..ننتبه إلى عدم انهاء التعليمات بالفاصلة المنقوطة ; ، ليس علينا سوى حفظ الملف ، ثم تنفيذه بكتابة / بعد محث SQL ..

- **أمر إظهار بنية جدول** Describe :

المقصود بهذا الأمر إظهار صفات حقول الجدول ، من حيث نوعها (رقم ، نص ، تاريخ ..الخ) وحجم هذا الحقل ، وبيان إن كان لهذا الحقل قيد أم لا ..وصيغة هذا الأمر ..

```
SQL> desc tablename;
```

حيثُ tablename هو اسم الجدول ..

فمثلاً إذا أردنا عرض خصائص الجدول emp .. نكتب الأمر التالي ..

```
SQL> desc emp;
```

والنتيجة هي كما يلي ..

```
Name                          Null?    Type
----------------------------- -------- ----
EMPNO                         NOT NULL NUMBER(4)
ENAME                                  VARCHAR2(10)
JOB                                    VARCHAR2(9)
MGR                                    NUMBER(4)
HIREDATE                               DATE
SAL                                    NUMBER(7,2)
COMM                                   NUMBER(7,2)
DEPTNO                                 NUMBER(2)
```

ملخص الدرس :

- من أوامر التحرير في بيئة SQL * Plus 3.3 في بيئة :-

- أمر الإضافة : ماسنضيفه sql>A

- أمر التعديل : sql> C / old /new

- حيثُ old الكلمة المُراد استبدالها ، و new الكلمة الجديدة .

- أمر مسح دارئ (مخزن مؤقت) SQL :

```
SQL> cl buff ;
```

- من أوامر التحرير في بيئة SQL * Plus 3.3 في بيئة :-

- أمر حفظ الملفات :

```
sql>sav filename.sql
```

حيثُ filename.sql اسم الملف المراد الحفظ به ، ويلزم بالإمتداد .. sql

- أمر جلب محتوى ملف :

```
sql>get filename.sql
```

حيثُ filename.sql اسم آخر ملف تم حفظه ضمن ذاكرة SQL المؤقتة ..

- أمر تنفيذ محتوى ملف :

```
sql>star filename.sql
```

أو الأمر ..

```
sql> @ filename.sql
```

حيثُ filename.sql اسم ملف محفوظ سابقاً ..

- أمر تشغيل برنامج تحرير ..

```
SQL> ed
```

- أمر عرض خصائص جدول ..

```
SQL> desc tablename;
```

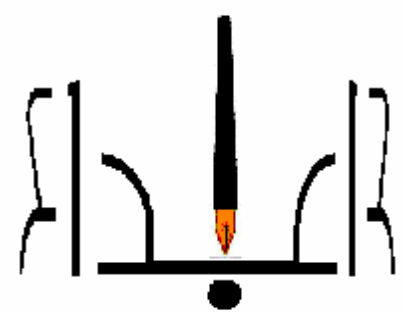
حيثُ tablename هو اسم الجدول ..

****

وفي الدرس القادم بإذن اللّه سنبدأ بجلمة الإستعلام .. Select

فإلى لقاء قريب بإذن اللّه تعالى .وسامحونا على التقصير ، وفق اللّه الجميع لما يحبه ويرضاه ..

الأسيـــــــــــــــــــــــف



Email:amaar1422@hotmail.com

بسم الله الرحمن الرحيم

سلسلة تعلم ORACLE بسهولة

الدرسُ الرابع — إنشاء الجداول في بيئة SQL

مقدمة :

كما علمنا سابقاً أن من لغات SQL الفرعية لغة D.D.L ، وهي لغة تعريف البيانات ، وتعتبر أصلاً بناء SQL ، وعندما تقوم بتعريف هذه البيانات يتم وضع إدخالات لها في قاموس البيانات الخاص بـ ORACLE ، وكما ذكرنا سابقاً أيضاً فإن الأوامر الأساسية لهذه اللغة هي :

- يستخدم لإنشاء الجداول (CREATE TABLE) .
- يستخدم للتعديل على جدول منشأ سابقاً (ALTER TABLE) .
- يستخدم لحذف جدول غير مرغوب فيه (DROP TABLE) .

وسنبدأ في درسنا هذا بشرح الأمر الأول والأساسي ..

- **أمر إنشاء الجداول** CREATE TABLE :

هناك نوعين من الأوامر لإنشاء جدول :

- انشاء جدول جديد (new table) .
- انشاء جدول جديد ، بصفات بحقول أخرى من جدول آخر قديم (copied table) .

١- إنشاء جدول جديد (create new table):

لكي نقوم بإنشاء جدول نستخدم الأمر CREATE كالتالي :

```
CREATE TABLE اسم الجدول ( حقول الجدول );
```

ويشترطُ في اسم الجدول عدة شروط وهي :

١- لا يتجاوز طول اسم الجدول عن 30 حرفاً .

٢- يمكن ان يكون اسم الجدول خليط من الأرقام والحروف والرموز الخاصه ولكن لابد أن يبدأ بحرف على الأقل .

٣- أن لا يكون اسم الجدول كلمة محجوزة في اللغة .

والشروط السابقة تنطبق أيضاً على أسماء الحقول في الجدول .

ويجب أن نعلم أن لحقول الجدول صفتين هما :

- طول هذا الحقل : أي الحجم الذي سيخزنه في قاعدة البيانات .
- القيود على هذا الحقل : وهي تعني الشروط اللازمة لقيم هذه الحقول ، وسنتحدث عنها لاحقاً بإذن الله .

كما أن للمتغيرات في لغة SQL عدة أنواع نذكرها مع القيمة القصوى لكل حقل ، علما بأن هذه القيم خاصة بـ ORACLE 8 ، ومن البديهي أن تزيد هذه القيم في الإصدارات الحديثة ..

١- بيانات حرفية CHAR :

ويستخدم هذا النوع لتخزين عدد ثابت من الحروف ، والحد الأقصى لعدد الأحرف هو 2000 بايت .

مثال :

اسم الحقل CHAR (16)

٢- بيانات حرفية كبيرة وهي على صورتين :

- var char : يستخدم هذا النوع لتخزين بيانات حرفية متنوعة ، والحد الأقصى لعدد الأحرف هو 4000 بايت .

مثال :

اسم الحقل VARCHAR (50)

- var char2 : يستخدم هذا النوع لتخزين بيانات حرفية متنوعة ، والحد الأقصى لعدد الأحرف هو 4000 بايت .

مثال :

اسم الحقل VARCHAR2 (50)

لكن ماهو الفرق بين VARCHAR & VARCHAR2 ؟

أن var char2 يسمى المتغير المطاطي أي لو حجزنا ١٠ خانات وكان الاسم يتكون من ٦ خانات فانه سوف يقصر الى ٦ خانات تلقائيا بعكس الـ var char فسوف يحجز جميع الخانات حتى ولو لم تستعمل .

٣-الحقل ذو القيمة الرقمية الصحيحة NUMBER :

تتكون البيانات المدخلة في هذا الحقل من الأرقام (٠، ١، ٢..... ٩) وتحديد طول الحقل اختياري ..

مثال :

اسم الحقل NUMBER (50)

٤-الحقل ذو القيمة الرقمية الحقيقية [ذو الفاصلة العشرية] NUMBER :

تتكون البيانات المدخلة في هذا الحقل من الأرقام (٠، ١، ٢، ..... ٩)

اسم الحقل NUMBER (I , j)

حيث يمثل I طول العدد العشري كاملاً شاملاً العدد الصحيح وما على يمين الفاصلة أيضاً ، أما j فيمثل طول الأعداد العشرية يمين الفاصلة .

٥- الحقل ذو القيمة الثنائية RAW :

يُستخدم لخزن بيانات ثنائية ، وأقصى طول له هو 2000 بايت .

٦- الحقل ذو قيمة تاريخية DATE :

ويُستخدم لخزن بيانات من نوع التاريخ (يوم ، شهر ، سنة) .

٧- الحقل ذو البيانات الكبيرة LONG :

ويُستخدم لخزن البيانات النصية ، والتي يصل طولها إلى 2 جيجا بايت .

وهناك أنواع أخرى لسنا بحاجة إليها الآن وهي :

- LONG RAW (يحتوي على بيانات ثنائية يصل طولها إلى 2 جيجا بايت) .
- ROWID (يحتوي على مواقع أسطر الجدول في القرص) .

وأما هذه الأربعة الأخيرة فهي موجودة فقط في الإصدار ٨ أو الإصدارات الأحدث منه :

- BLOB (كائن ثنائي كبير) .
- CLOB (كائن كبير يعتمد على المحارف) .
- NCLOB (كائن كبير يعتمد على المحارف وحيدة البايت أو متعددة البياتات).
- BFILE (ملف خارجي كبير) .

والآن سنتعرف على طريقة إنشاء جدول جديد ، بإعطاء حقوله تعاريف من المتغيرات السابقة ..

ولكن قبل أن نكتب أي تعليمة ، يجب أن تكتبها تحت اسم المستخدم الخاص بك ، فإذا كنت تحت أي مستخدم آخر ، فاخرج منه ، وادخل باسمك ..

مثلاً : نريد القيام بإنشاء جدول للمدارس باسم SCHOOL ويحتوي على الحقول التالية :

- رقم المدرسة ونوعه رقمي بطول 10 مفتاح أساسي .
- اسم المدرسة ونوعه حرفي بطول 30 غير فارغ .
- نوع المدرسة (ابتدائي ، متوسط ، ثانوي) ونوعه حرفي بطول 15 غير فارغ .
- موقع المدرسة ونوعه حرفي بطول 30 .
- تاريخ تأسيس المدرسة ونوعه تاريخ .

و سنكتب تعليمات إنشاء جدول جديد بالطريقة التالية ، وهذا عند كل إنشاء جدول ..

```
CREATE   TABLE   SCHOOL
(
    اسم الحقل    نوع البيانات  (طولها)  [ فارغ أو غير فارغ أو قيود ] ,
    اسم الحقل    نوع البيانات  (طولها)  [ فارغ أو غير فارغ أو قيود ] ,
    ..... ,
    ..... ,
    ..... ,
    اسم الحقل    نوع البيانات  (طولها)  [ فارغ أو غير فارغ أو قيود ]
);
```

وماهو مكتوبٍ بالأحمر هي القيود على هذه الحقل ..وسنشرح كيفية إنشاء القيود على الحقول في درسنا القادم بإذن الله ..

إذاً نكتب ما يلي لإنشاء جدول المدرسة ..

```
CREATE  TABLE  SCHOOL
(
   S_NO      NUMBER  (10) ,
   S_NAME  VARCHAR 2 (30) ,
   S_TYPE    CHAR  (15) ,
   S_LOC     VARCHAR 2  (30) ,
   S_DATE    DATE
) ;
```

وإذا ظهرت لنا العبارة التالية ، نكون قد نجحنا في إنشاء الجدول ..

```
Table created .
```

ويفضل أن يكون اسم الحقل على جزأين :

- جزأ يشير إلى اسم الجدول ، وهنا أخذنا الحرف S ليشير إليه .
- جزأ يشير إلى اسم الحقل نفسه ويكون معبراً عنه .

وهذا التفضيل سببه التسهيل ، وذلك عند الإستعلام عن الحقول أو إجراء بعض العمليات عليها ..

كما اقتصرنا عند إنشاء الجدول السابق على تعريف الحقول بالأطوال المناسبة التي ستأخذها ، ولم نكتب أي قيد على أي حقل ..حتى نتعلمها في درسنا القادم بإذن الله .

وبعدأن نجحنا في إنشاء جدول ، لابد من معرفة كيفية استعراض مواصفات الجدول بعد إنشائه أو أي جداول أخرى ... وذلك بكتابة الأمر التالي :

```
SQL> DESC SCHOOL;
```

وستظهر لنا مواصفات جدول المدرسة كمايلي ..

```
Name                          Null?    Type
----------------------------- -------- ----
S_NO                                   NUMBER(10)
S_NAME                                 CHAR2(30)
S_TYPE                                 CHAR(15)
S_LOC                                  CHAR2(30)
S_DATE                                 DATE
```

ونختم درسنا هذا ببيان الطريقة الثانية لإنشاء الجداول ..

٢- إنشاء جدول جديد ، بصفات بحقول أخرى من جدول آخر قديم (copied table) :

نقوم هنا بإنشاء جدول جديد يحوي بعض الحقول ، بدون أن نعرفها من ناحية الطول ، وذلك بصفات حقول أخرى نختارها من جدول آخر ، ولكي نقوم بذلك نستخدم الصيغة التالية :

```
CREATE   TABLE    اسم الجدول الجديد  ( حقول الجدول الجديد )
AS
SELECT  ( نختار حقول من الجدول القديم )
FROM    اسم الجدول القديم   ;
```

مثال :

سنقوم بإنشاء جدول جديد اسمه STUDENT ، ونتخار له بعض الحقول من من الجدول السابق الذي أنشأناه وهو SCHOOL ، وسنختار حقلين منه ..

```
SQL> CREATE TABLE STUDENT (S_NO,S_NAME)
  AS
   SELECT    S_NAME,S_TYPE
   FROM  SCHOOL ;

Table created .
```

نلاحظ من التعليمات السابقة أننا أنشأنا حقلي جدول الطلاب المسمى STUDENT ، ويحوي على حقلين هما رقم الطالب (S_NO) واسم الطالب (S_NAME) ، بدون أن نعرف أطوالهما ، وذلك لأننا أخذنا أطوالهما بأخذنا حقلين هما حقل اسم المدرسة (S_NAME) ونوع المدرسة (S_TYPE) من الجدول السابق SCHOOL.

ولكي تتأكد من كلامي السابق ، اكتب أمر استعراض خصاءئص الجدول الجديد STUDENT
وستجد أنه أعطى حقوله خصائص الحقلين المختارين من جدول SCHOOL ، كما يلي ..

```
SQL> DESC  STUDENT ;

 Name                      Null?    Type
 ------------------------- -------- ----
 S_NO                               CHAR(30)
 S_NAME                             CHAR(15)
```

ملخص الدرس :

* *هناك نوعين من الأوامر لإنشاء جدول :*

- انشاء جدول جديد وصيغته :

```
CREATE   TABLE  اسم الجدول  ( حقول الجدول );
```

- انشاء جدول جديد وبصفات حقول أخرى من جدول آخر قديم وصيغته :

```
CREATE   TABLE     اسم الجدول الجديد  ( حقول الجدول الجديد )
AS
SELECT  ( نختار حقول من الجدول القديم )
FROM    اسم الجدول القديم   ;
```

* *ويشترطُ في اسم الجدول وحقوله عدة شروط وهي :*

١- لا يتجاوز طول اسم الجدول عن 30 حرفاً .

٢- يمكن ان يكون اسم الجدول خليط من الأرقام والحروف والرموز الخاصه ولكن لابد أن يبدأ بحرف على الأقل .

٣- أن لا يكون اسم الجدول كلمة محجوزة في اللغة .

ولحقول الجدول صفتين هما :

- طول هذا الحقل : أي الحجم الذي سيخزنه في قاعدة البيانات .
- القيود على هذا الحقل : وهي تعني الشروط اللازمة لقيم هذه الحقول

- *للمتغيرات في لغة SQL عدة أنواع أهمها ما يلي :*

CHAR و VARCHAR و VARCHAR2 و NUMBER و RAW و DATA و LONG .

*الفرق بين VARCHAR & VARCHAR2 :

أنّ VARCHAR2 يسمى المتغير المطاطي أي لو حجزنا ١٠ خانات وكان الاسم يتكون من ٦ خانات فانه سوف يقصر الى ٦ خانات تلقائيا بعكس الـ VARCHAR فسوف يحجز جميع الخانات حتى ولو لم تستعمل .

****

وفي الدرس القادم بإذن الله سنشرح ماهية القيود وأنواعها مع مثال لكل نوع ..

فإلى لقاء قريب بإذن الله تعالى .وسامحونا على التقصير ، وفق الله الجميع لما يحبه ويرضاه ..

الأسيـــــــــــــــــــــــف

Email:amaar1422@hotmail.com

بسم الله الرحمن الرحيم

سلسلة تعلم ORACLE بسهولة

الدرسُ الرابع — إنشاء الجداول في بيئة SQL

مقدمة :

كما علمنا سابقاً أن من لغات SQL الفرعية لغة D.D.L ، وهي لغة تعريف البيانات ، وتعتبر أصلاً بناء SQL ، وعندما تقوم بتعريف هذه البيانات يتم وضع إدخالات لها في قاموس البيانات الخاص بـ ORACLE ، وكما ذكرنا سابقاً أيضاً فإن الأوامر الأساسية لهذه اللغة هي :

- يستخدم لإنشاء الجداول (CREATE TABLE) .
- يستخدم للتعديل على جدول منشأ سابقاً (ALTER TABLE) .
- يستخدم لحذف جدول غير مرغوب فيه (DROP TABLE) .

وسنبدأ في درسنا هذا بشرح الأمر الأول والأساسي ..

- **أمر إنشاء الجداول** CREATE TABLE :

هناك نوعين من الأوامر لإنشاء جدول :

- انشاء جدول جديد (new table) .
- انشاء جدول جديد ، بصفات بحقول أخرى من جدول آخر قديم (copied table) .

١- إنشاء جدول جديد (create new table):

لكي نقوم بإنشاء جدول نستخدم الأمر CREATE كالتالي :

```
CREATE TABLE اسم الجدول ( حقول الجدول );
```

ويشترطُ في اسم الجدول عدة شروط وهي :

١- لا يتجاوز طول اسم الجدول عن 30 حرفاً .

٢- يمكن ان يكون اسم الجدول خليط من الأرقام والحروف والرموز الخاصه ولكن لابد أن يبدأ بحرف على الأقل .

٣- أن لا يكون اسم الجدول كلمة محجوزة في اللغة .

والشروط السابقة تنطبق أيضاً على أسماء الحقول في الجدول .

ويجب أن نعلم أن لحقول الجدول صفتين هما :

- طول هذا الحقل : أي الحجم الذي سيخزنه في قاعدة البيانات .
- القيود على هذا الحقل : وهي تعني الشروط اللازمة لقيم هذه الحقول ، وسنتحدث عنها لاحقاً بإذن الله .

كما أن للمتغيرات في لغة SQL عدة أنواع نذكرها مع القيمة القصوى لكل حقل ، علما بأن هذه القيم خاصة بـ ORACLE 8 ، ومن البديهي أن تزيد هذه القيم في الإصدارات الحديثة ..

١- بيانات حرفية CHAR :

ويستخدم هذا النوع لتخزين عدد ثابت من الحروف ، والحد الأقصى لعدد الأحرف هو 2000 بايت .

مثال :

اسم الحقل CHAR (16)

٢- بيانات حرفية كبيرة وهي على صورتين :

- var char : يستخدم هذا النوع لتخزين بيانات حرفية متنوعة ، والحد الأقصى لعدد الأحرف هو 4000 بايت .

مثال :

اسم الحقل VARCHAR (50)

- var char2 : يستخدم هذا النوع لتخزين بيانات حرفية متنوعة ، والحد الأقصى لعدد الأحرف هو 4000 بايت .

مثال :

اسم الحقل VARCHAR2 (50)

لكن ماهو الفرق بين VARCHAR & VARCHAR2 ؟

أن var char2 يسمى المتغير المطاطي أي لو حجزنا ١٠ خانات وكان الاسم يتكون من ٦ خانات فانه سوف يقصر الى ٦ خانات تلقائيا بعكس الـ var char فسوف يحجز جميع الخانات حتى ولو لم تستعمل .

٣-الحقل ذو القيمة الرقمية الصحيحة NUMBER :

تتكون البيانات المدخلة في هذا الحقل من الأرقام (٠، ١، ٢..... ٩) وتحديد طول الحقل اختياري ..

مثال :

اسم الحقل NUMBER (50)

٤-الحقل ذو القيمة الرقمية الحقيقية [ذو الفاصلة العشرية] NUMBER :

تتكون البيانات المدخلة في هذا الحقل من الأرقام (٠، ١، ٢، ..... ٩)

اسم الحقل NUMBER (I , j)

حيث يمثل I طول العدد العشري كاملاً شاملاً العدد الصحيح وما على يمين الفاصلة أيضاً ، أما j فيمثل طول الأعداد العشرية يمين الفاصلة .

٥- الحقل ذو القيمة الثنائية RAW :

يُستخدم لخزن بيانات ثنائية ، وأقصى طول له هو 2000 بايت .

٦- الحقل ذو قيمة تاريخية DATE :

ويُستخدم لخزن بيانات من نوع التاريخ (يوم ، شهر ، سنة) .

٧- الحقل ذو البيانات الكبيرة LONG :

ويُستخدم لخزن البيانات النصية ، والتي يصل طولها إلى 2 جيجا بايت .

وهناك أنواع أخرى لسنا بحاجة إليها الآن وهي :

- LONG RAW (يحتوي على بيانات ثنائية يصل طولها إلى 2 جيجا بايت) .
- ROWID (يحتوي على مواقع أسطر الجدول في القرص) .

وأما هذه الأربعة الأخيرة فهي موجودة فقط في الإصدار ٨ أو الإصدارات الأحدث منه :

- BLOB (كائن ثنائي كبير) .
- CLOB (كائن كبير يعتمد على المحارف) .
- NCLOB (كائن كبير يعتمد على المحارف وحيدة البايت أو متعددة البياتات).
- BFILE (ملف خارجي كبير) .

والآن سنتعرف على طريقة إنشاء جدول جديد ، بإعطاء حقوله تعاريف من المتغيرات السابقة ..

ولكن قبل أن نكتب أي تعليمة ، يجب أن تكتبها تحت اسم المستخدم الخاص بك ، فإذا كنت تحت أي مستخدم آخر ، فاخرج منه ، وادخل باسمك ..

مثلاً : نريد القيام بإنشاء جدول للمدارس باسم SCHOOL ويحتوي على الحقول التالية :

- رقم المدرسة ونوعه رقمي بطول 10 مفتاح أساسي .
- اسم المدرسة ونوعه حرفي بطول 30 غير فارغ .
- نوع المدرسة (ابتدائي ، متوسط ، ثانوي) ونوعه حرفي بطول 15 غير فارغ .
- موقع المدرسة ونوعه حرفي بطول 30 .
- تاريخ تأسيس المدرسة ونوعه تاريخ .

و سنكتب تعليمات إنشاء جدول جديد بالطريقة التالية ، وهذا عند كل إنشاء جدول ..

```
CREATE   TABLE   SCHOOL
(
    اسم الحقل    نوع البيانات  (طولها)  [ فارغ أو غير فارغ أو قيود ] ,
    اسم الحقل    نوع البيانات  (طولها)  [ فارغ أو غير فارغ أو قيود ] ,
    ..... ,
    ..... ,
    ..... ,
    اسم الحقل    نوع البيانات  (طولها)  [ فارغ أو غير فارغ أو قيود ]
);
```

وماهو مكتوبٍ بالأحمر هي القيود على هذه الحقل ..وسنشرح كيفية إنشاء القيود على الحقول في درسنا القادم بإذن الله ..

إذاً نكتب ما يلي لإنشاء جدول المدرسة ..

```
CREATE  TABLE  SCHOOL
(
   S_NO      NUMBER  (10) ,
   S_NAME  VARCHAR 2 (30) ,
   S_TYPE    CHAR  (15) ,
   S_LOC     VARCHAR 2  (30) ,
   S_DATE    DATE
) ;
```

وإذا ظهرت لنا العبارة التالية ، نكون قد نجحنا في إنشاء الجدول ..

```
Table created .
```

ويفضل أن يكون اسم الحقل على جزأين :

- جزأ يشير إلى اسم الجدول ، وهنا أخذنا الحرف S ليشير إليه .
- جزأ يشير إلى اسم الحقل نفسه ويكون معبراً عنه .

وهذا التفضيل سببه التسهيل ، وذلك عند الإستعلام عن الحقول أو إجراء بعض العمليات عليها ..

كما اقتصرنا عند إنشاء الجدول السابق على تعريف الحقول بالأطوال المناسبة التي ستأخذها ، ولم نكتب أي قيد على أي حقل ..حتى نتعلمها في درسنا القادم بإذن الله .

وبعدأن نجحنا في إنشاء جدول ، لابد من معرفة كيفية استعراض مواصفات الجدول بعد إنشائه أو أي جداول أخرى ... وذلك بكتابة الأمر التالي :

```
SQL> DESC SCHOOL;
```

وستظهر لنا مواصفات جدول المدرسة كمايلي ..

```
Name                          Null?    Type
----------------------------- -------- ----
S_NO                                   NUMBER(10)
S_NAME                                 CHAR2(30)
S_TYPE                                 CHAR(15)
S_LOC                                  CHAR2(30)
S_DATE                                 DATE
```

ونختم درسنا هذا ببيان الطريقة الثانية لإنشاء الجداول ..

٢- إنشاء جدول جديد ، بصفات حقول أخرى من جدول آخر قديم (copied table) :

نقوم هنا بإنشاء جدول جديد يحوي بعض الحقول ، بدون أن نعرفها من ناحية الطول ، وذلك بصفات حقول أخرى نختارها من جدول آخر ، ولكي نقوم بذلك نستخدم الصيغة التالية :

```
CREATE   TABLE    اسم الجدول الجديد  ( حقول الجدول الجديد )
AS
SELECT  ( نختار حقول من الجدول القديم )
FROM   اسم الجدول القديم   ;
```

مثال :

سنقوم بإنشاء جدول جديد اسمه STUDENT ، ونختار له بعض الحقول من من الجدول السابق الذي أنشأناه وهو SCHOOL ، وسنختار حقلين منه ..

```
SQL> CREATE TABLE STUDENT (S_NO,S_NAME)
  AS
   SELECT    S_NAME,S_TYPE
   FROM  SCHOOL ;

Table created .
```

نلاحظ من التعليمات السابقة أننا أنشأنا حقلي جدول الطلاب المسمى STUDENT ، ويحوي على حقلين هما رقم الطالب (S_NO) واسم الطالب (S_NAME) ، بدون أن نعرف أطوالهما ، وذلك لأننا أخذنا أطوالهما بأخذنا حقلين هما حقل اسم المدرسة (S_NAME) ونوع المدرسة (S_TYPE) من الجدول السابق SCHOOL.

ولكي تتأكد من كلامي السابق ، اكتب أمر استعراض خصاءئص الجدول الجديد STUDENT
وستجد أنه أعطى حقوله خصائص الحقلين المختارين من جدول SCHOOL ، كما يلي ..

```
SQL> DESC  STUDENT ;

 Name                      Null?    Type
 ------------------------- -------- ----
 S_NO                               CHAR(30)
 S_NAME                             CHAR(15)
```

ملخص الدرس :

* *هناك نوعين من الأوامر لإنشاء جدول :*

- انشاء جدول جديد وصيغته :

```
CREATE   TABLE  اسم الجدول  ( حقول الجدول );
```

- انشاء جدول جديد وبصفات حقول أخرى من جدول آخر قديم وصيغته :

```
CREATE   TABLE     اسم الجدول الجديد  ( حقول الجدول الجديد )
AS
SELECT  ( نختار حقول من الجدول القديم )
FROM    اسم الجدول القديم   ;
```

* *ويشترطُ في اسم الجدول وحقوله عدة شروط وهي :*

١- لا يتجاوز طول اسم الجدول عن 30 حرفاً .

٢- يمكن ان يكون اسم الجدول خليط من الأرقام والحروف والرموز الخاصه ولكن لابد أن يبدأ بحرف على الأقل .

٣- أن لا يكون اسم الجدول كلمة محجوزة في اللغة .

ولحقول الجدول صفتين هما :

- طول هذا الحقل : أي الحجم الذي سيخزنه في قاعدة البيانات .
- القيود على هذا الحقل : وهي تعني الشروط اللازمة لقيم هذه الحقول

- *للمتغيرات في لغة SQL عدة أنواع أهمها ما يلي :*

CHAR و VARCHAR و VARCHAR2 و NUMBER و RAW و DATA و LONG .

*الفرق بين VARCHAR & VARCHAR2 :

أنّ VARCHAR2 يسمى المتغير المطاطي أي لو حجزنا ١٠ خانات وكان الاسم يتكون من ٦ خانات فانه سوف يقصر الى ٦ خانات تلقائيا بعكس الـ VARCHAR فسوف يحجز جميع الخانات حتى ولو لم تستعمل .

****

وفي الدرس القادم بإذن الله سنشرح ماهية القيود وأنواعها مع مثال لكل نوع ..

فإلى لقاء قريب بإذن الله تعالى .وسامحونا على التقصير ، وفق الله الجميع لما يحبه ويرضاه ..

الأسيـــــــــــــــــــــــف

Email:amaar1422@hotmail.com

بسم الله الرحمن الرحيم

سلسلة تعلم ORACLE بسهولة

الدرسُ الخامس : إنشاء القيود على حقول الجدوال

قبل أن نبدأ درسنا هذا أننبه إلى ضرورة تطبيقكم لكل صغيرة وكبيرة فيه ..ولكن ان تستطيعو بناء جدوال باسمكم الخاص ، إلا بعد إعطائه الصلاحية ..

الآن ليس لكم إلاصلاحية الإتصال فقط ، ولكي تكون ليكم باقي الصلاحيات ، نكتب الأمر التالي ..

SQL> grant resource to **إسم المستخدم** ;

فتظهر لنا العبارة التالية ، والتي تخبرنا أنّه تم منح الصلاحية ..

Grant succeeded.

والآن بإمكانكم إنشاء الجدوال والتعامل معها بكل حرية من خلال اسمك الخاص ..

قمنا في درسنا السابق بإنشاء جدول يحوي عدة حقول بمواصفاتٍ معينة ، ولكنها لاتحمل أي قيودٍ أو شروط لوضع البيانات بداخلها ، ووضع هذه القيود على الأعمدة في غاية الأهمية من أجل وضع شروط مناسبة وواضحة على قيم هذه الأعمدة ..

*** هناك مجموعة قيود يمكن أن نستخدمها على الحقول وهي كالتالي :**

- يجب إدخال قيمة في الحقل Nut Null .
- أن لا تكرر قيمة الحقل Unique .
- وضع قيمة افتراضية للحقل Default .
- إجراء فحص معين على الحقل Check .
- إنشاء قيد مفتاح رئيسي Primary Key .
- إنشاء قيد مفتاح ثانوي Foreign Key .

* أما كيفية إنشاء القيود على حقول الجداول ...فهناك طريقتان متبعتان في ذلك :

١- **الطريقة الأولى** In line constraint :

وتعني إنشاء قيود على مستوى **تعريف الحقل** ، أي يُكتب القيد في نفس سطر تعريف الحقل ، أو في السطر التالي مباشرة ..

٢- **الطريقة الثانية** Out line constraint :

وتعني إنشاء قيود على مستوى **تعريف الجدول** ، أي تُكتب جميع القيود بعد الإنتهاء من تعريف الحقول ، وهذه الطريقة هي المعتمدة من الشركة ، وهي الأفضل حسب رأي الكثيرين ..

وننبه هنا إلى أن هاتين الطريقتين تسريان على جميع القيود الستة التي ذكرناها ، ماعدا القيد ذو القيمة الإفتراضية للحقل (Default) فتسري عليه الطريقة الأولى فقط .

وهناك شرطان على اسم القيد الذي نكتبه وهما :

- أن لايتكرر اسم القيد .
- أن لايزيد عن 30 حرف .

وسنبدأ بشرح هذه القيود إن شاء الله ، مع ذكر مثال لكل قيد بطريقتي كتابته ..

١- القيـــدُ الأول (Nut Null) :

وهو وضع قيد على حقلٍ ما ، بحيث لايكون هذا الحقل ذو قيمة فارغة ..أي يجب أن يحتوي على قيمة .. كحقل رقم الطالب وحقل اسم الطالب مثلاً ، يجب أن يحويان على قيمة ..

- **مثال بطريقة** In line :

```
SQL> create table customer (
  2  cust_no number(4) not null,
  3  cust_name varchar2(40)
     not null,
  4  cust_address varchar2 (30)
  5  );
```

نلاحظ هنا :

- أنشأنا هنا الجدول الخاص بالزبائن ؛ أن الحقل الأول : رقم الزبون (cust_no) والحقل الثاني : اسم الزبون (cust_name) كلاهما عرفناهما بأن لايكونا ذو قيمة فارغة ..لأنه يجب معرفة رقم الزبون واسمه ..أما الحقل الثالث : عنوان الزبون (cust_address)، فليس مهماً ، فلا ضير أن يكون فارغاً عند إدخال البيانات .
- أننا ذكرنا اسم القيد وهو not null بعد تعريف الحقل مباشرة ، سواءً في نفس السطر كما في تعريف حقل رقم الزبون ، أو في سطرٍ آخر كما في تعريف حقل اسم الزبون .

- **مثال بطريقة** Out line :

```
SQL> create table customer (
  2  cust_no number(4),
  3  cust_name varchar2(40) ,
  4  cust_address varchar2 (30),
  5  constraint cust_no_nt
  6  check (cust_no is not null),
  7  constraint cust_name_nt
  8  check ( cust_name  is not null)
  9  );
```

*** إما هنا فنلاحظ :**

أننا عرفنا الحقول أولاً ، ثم وضعنا القيود ، وهذه الطريقة أفضل من الأولى ؛ فمثلاً عندما عرفنا الحقل (رقم الزبون) كتبنا الآتي ..

constraint cust_no_nt

حيثُ constraint **كلمة محجوزة معناها قيد ، ثم نذكر اسم القيد وهو يتكون من جزأين :**

- **الجزءُ الأول .. هو اسم الحقل نفسه ..**
- **الجزءُ الثاني .. هو اختصار لاسم القيد ، وهنا اختصرناه بـ**nt **وبهذه الطريقة نكتب اسماء بقية القيود الأخرى ..**
- **أما** check (cust_no is not null) **فمعناه افحص الحقل** cust_no **هل ليس فارغاً ..وهذا ما نريده أن لا يكون فارغاً (أي يحتوي على قيمة) .**

٢- القيــدُ الثاني (Unique) :.

ومعناه ألا تتكرر قيمة هذا الحقل الذي سنضع هذا القيد عليه ، فمثلاً لو وضعنا هذا القيد على حقل رقم الزبون ، نستنتج أنه لا نريد أن يتكرر رقم الزبون ..كما يلي :

- **مثال بطريقة** In line :

```
SQL> create table customer (
  2  cust_no number(4) unique,
  3  cust_name varchar2(40) not null,
  4  cust_address varchar2(30)
  5  );
```

- **مثال بطريقة** Out line :

```
SQL> create table customer (
  2  cust_no number(4),
  3  cust_name varchar2(40) not null,
  4  cust_address varchar2(30),
  5  constraint cust_no_uni
  6  unique ( cust_no)
  7  );
```

٣- القيـــدُ الثالث (Default) :.

ونستفيد منه في وضع قيمة افتراضية لحقلٍ ما ، مثلاً لحقل عمر الطالب ، نضع قيمة افتراضية لعمره ، وذلك في حالة عدم إدخال المستخدم أي قيمة ..

وهذا القيد هو الوحيد الذي يُكتب بطريقة In line فقط ..

مثال :

```
create table student (
std_no number(7) not null,
std_name varchar2(40) not null,
std_age number(2)
default 20 ,
std_nation varchar2(20)
default 'saudi'
);
```

ونلاحظ هنا :

- أننا عرفنا حقل عمر الطالب (std_no) على أنه رقم وأعطيناه قيمةً افتراضية في حالة عدم إدخال المستخدم لأي قيمة ، وهي القيمة 20 سنة .
- و عرفنا حقل جنسية الطالب (std_nation)على أنه نصي ، وأعطيناه قيمةً افتراضية في حالة عدم إدخال المستخدم لأي قيمة ، وهي الجنسية Saudi .

٤- القيـــدُ الرابـــع (Check) :.

ونستفيد منه عندما نريد أن نفحص قيمة مدخلة لحقل معين يقبل مجموعة قيم محددة سلفاً، حيث يقوم القيد بفحص القيمة المخلة من بين القيم الموجودة .

- **مثال بطريقه** In line :

```
SQL> create table student (
  2  std_no number(7) not null,
  3  std_name varchar2(40) not null,
  4  std_sex varchar2(1)
  5  check ( std_sex in (' m ',' f ')),
  6  std_case varchar2(1)
  7  check ( std_case in (' s ',' m',' w ',' d ')),
  8  std_age number(2)
  9  check ( std_age between 19 and 30 )
 10  );
```

ونلاحظ هنا:

- **أننا عرفنا حقل جنس الطالب** (std_sex) **على أنه نصي وأعطيناه قيد الفحص** check .. **فهنا يفحص قيمة الحقل ، فعندما يدخل المستخدم الحرف** m **يعني ذلك أن الطالب ذكر .. وعندما يدخل المستخدم الحرف** f **يعني ذلك أن الطالب أنثى ..**
- **وعرفنا حقل الحالة الإجتماعية للطالب** (std_case) **على أنه نصي وأعطيناه قيد الفحص** check ..**وهنا يفحص قيمة الحقل ، فعندما يدخل المستخدم الحرف فهنا يفحص قيمة الحقل ..**
  - **فعندما يدخل المستخدم الحرف** s **يعني ذلك أن الطالب متزوج .**
  - **وعندما يدخل المستخدم الحرف** w **يعني ذلك أن الطالب أرمل .**
  - **وعندما يدخل المستخدم الحرف** m **يعني ذلك أن الطالب مطلق .**
  - **وعندما يدخل المستخدم الحرف** d **يعني ذلك أن الطالب أعزب .**
- **وعرفنا حقل عمر الطالب** (std_ age) **على أنه نصي وأعطيناه قيد الفحص** check ..**وهنا يفحص قيمة الحقل مابين القيمة العمرية من** 19 **إلى** 30 ..

- **المثال بطريقة** Out line :

```
SQL> create table student
  2 ( std_no number(7) not null,
  3  std_name varchar2(40) not null,
  4  std_sex varchar2(1),
  5  std_case varchar2(1),
  6  std_age number(2),
  7  constraint std_sex_chk
  8  check (std_sex in (' m ',' f ')),
  9  constraint std_case_chk
 10 check( std_case in (' s ',' m',' w ',' d ')),
 11 constraint stdstd_age_chk
 12 check  ( std_age between 19 and 30)
 13  );
```

وأخيرا ..رجـــاء حـــــار إليكم ..

تطبيق كل مثال أورده هنا مع كل قيد وبكلا الطريقتين ،و بدون نسخ الجدول ولصقه ، قم بكتابة كل تعليمة بنفسك حتى تستفيد وتتعلم ..

اكتب الجدول بالطريقة الأولى ثم قم بحذفه بالأمر التالي ..

SQL> drop table **اسم الجدول** ;

ثم اكتب الجدول بالطريقة الثانية وهكذا ...

ملخص الدرس :

* *أمر إعطاء صلاحية إنشاء الجداول وغيرها من الكائنات في SQL :*

SQL> grant resource to **اسم المستخدم** ;

فتظهر لنا العبارة التالية ، والتي تخبرنا أنّه تم منح الصلاحية ..

Grant succeeded.

* *هناك مجموعة قيود يمكن أن نستخدمها على الحقول وهي كالتالي :*

- **يجب إدخال قيمة في الحقل** Nut Null .
- **أن لا تكرر قيمة الحقل** Unique .
- **وضع قيمة افتراضية للحقل** Default .
- **إجراء فحص معين على الحقل** Check .
- **إنشاء قيد مفتاح رئيسي** Primary Key .
- **إنشاء قيد مفتاح ثانوي** Foreign Key .

* *هناك طريقتان لإنشاء القيود على الحقول :*

- **على مستوى تعريف الحقل** In line constrain..
- **على مستوى تعريف الجدول** Out line constrain..

* *اسم القيد هو يتكون من جزأين :*

- **الجزءُ الأول .. هو اسم الحقل نفسه ..**
- **الجزءُ الثاني .. هو اختصار لإسم القيد ..**

وهاتين الطريقتين تسريان على جميع القيود الستة التي ذكرناها ، ماعدا القيد ذو القيمة الإبتدائية للحقل (Default) فتسري عليه الطريقة الأولى فقط .

- *أمر حذف جدول هو :*

SQL> drop table **اسم الجدول** ;

****

مراجعة مهمة على الدروس السابقة :

١ / قم بإنشاء مستخدم جديد :

- بالإسم Mohmad وبكلمة السر moh .
- امنحه صلاحية الإتصال بأوراكل ..
- امنحه صلاحية إنشاء الجدوال (من درس اليوم) ..

٢ / قم بإنشاء جدول بالمواصفات التالية :

- اسم الجدول : " الطلاب " ، وبالحقول التالية :
  - (رقم الطالب بحجم ٨ بايتات)
  - اسم الطالب بحجم ٥٠ بت .
  - تاريخ الميلاد .
  - معدل الطالب في الثانوية .
- اظهر مواصفات الجدول .
- احفظ نتيجة المواصفات في ملف .
- استدعي الملف المحفوظ .
- امسح مخزن ذاكرة SQL المؤقتة ..

الرجاء الإجابة عليها ، وبعث الإجابات على بريدي الخاص ، حتى نعلم مدى استيعابكم وفهمكم ..

وفي الدرس القادم بإذن اللّه سنكمل ما بقي لنا من موضوع القيود على الجدوال ، وهما من أهم القيود والمسؤلان عن ربط الجدوال بعضها ببعض ، ألا وهما : قيد المفتاح الرئيسي وقيد المفتاح الثانوي (الأجنبي) ..

فإلى لقاء قريب بإذن اللّه تعالى ...

وسامحونا على التأخر في طرح الدروس لظروف خارجة عن إرادتي ..

وفق اللّه الجميع لما يحبه ويرضاه ..

الأسيـــــــــــــــــــــــف

Email:amaar1422@hotmail.com

بسم الله الرحمن الرحيم

سلسلة تعلم ORACLE بسهولة

الدرسُ السادس تابع إنشاء القيود على حقول الجداول

قمنا في درسنا السابق بشرح الأربعة قيودِ الأولى ، وفي درسنا هذا اليوم القصير في مادته العلمية ..المهم في مضمونه ؛ سنكملُ إن شاء الله القيدين الأخيرين ، وهما الأهم من جميع القيود لأنهما المسؤلان عن ربط الجداول بعضها ببعض من حيث تكوين العلاقات بينها .. وهما : قيدَ المفتاح الرئيسي وقيدَ المفتاح الثانوي (الأجنبي) ..

٥-القيـــدُ الخامس (Primary Key) :

وظيفة هذا القيد إعطاء حقل معين من عدة حقول في جدولٍ ما ؛ صفة المفتاح الرئيسي في هذا الجدول ..

والمفتاح الرئيسي كما عرفناه في الدرس الثالث في دورة مفاهيم قواعد البيانات أنه :

المفتاح الذي يُحدد بشكل وحيد ومتفرد بحيث يتميز عن غيره ، فلا تتكرر قيمته في أكثر من حقل واحد ، ولا يقبل قيمة Null (أي لا يمكننا أن نترك الحقل فارغاً بدون قيمة) .

- مثال بطريقة In line :

```
SQL> create table student
  2  ( std_no number(10)
  3  primary key,
  4  std_name varchar2(50)
  5  );
```

نلاحظ هنا :

أننا أنشأنا جدول الطلاب بمفتاح رئيسي هو حقل رقم الطالب (std_no)، فنتعامل مع بيانات الطلاب ، بواسطة هذا الحقل والذي هو مفتاح رئيسي ؛ ونستطيع من خلاله البحث عن طالب معين

وكذلك عند حذف طالب معين ، بمجرد أن ندخل رقم الطالب ، ويكون هذا المفتاح رئيسياً في جدول الطلاب ، وأجنبياً في الجداول الأخرى ..

مثال بطريقة Out line :

```
SQL> create table student
  2  ( std_no number(10) ,
  3  std_name varchar2(50),
  4  constraint pk _std_01
  5  primary key (std_no)
  6  );
```

ملاحظة هامة جداً :

في بعض الحالات نظطر إلى إنشاء أكثر من حقل رئيسي في جدولٍ واحد كما في الـ Access وفي ORACLE نستطيع ذلك ولكن بالطريقة الثانية الغير خطية (Out line) .

٦- القيـــدُ السادس (Foreign Key) :

وظيفة هذا القيد إعطاء حقل معين من عدة حقول في جدولٍ ما ؛ صفة المفتاح الأجنبي في هذا الجدول .

والمفتاح الأجنبي كما عرفناه في الدرس الثالث في دورة مفاهيم قواعد البيانات أنه :

عبارة عن حقل (صفة) أو أكثر يستخدم للربط بين جدولين ، وسُمي المفتاح الأجنبي بهذا الاسم لأنه ليس من الحقول الموجودة أصلاً في الجدول ، أي أنه عبارة عن حقل أو أكثر تُضاف إلى جدول لربطه مع جدول آخر .

ويستخدم المفتاح الأجنبي كمؤشر مقابل للمفتاح الرئيسي ، بمعنى آخر فإن المفتاح الأجنبي هو عبارة عن حقل (صفة) أو أكثر تُضاف لجدول لربطه مع جدول آخر ، مع الالتزام بوجود مفتاح رئيسي مقابل مع ملاحظة أن المفتاح الأجنبي يجب أن يكون من نفس نوع بيانات (Data Type) المفتاح الرئيسي ، فلو كان المفتاح الرئيسي من النوع رقم مثلاً ، يجب أن يكون المفتاح الأجنبي من النوع رقم ، لذلك يعتبر الأجنبي مؤشر للرئيسي .

- **مثال بطريقة** In line :

حتى ننشئ مفتاحاً أجنبياً في جدول ما ، يجب أن يكون هذا المفتاح رئيسياً في جدولٍ آخر أنشأناهُ سابقاً ، وفي هذا المثال سنقوم إن شاء الله بإنشاء مفتاح رئيسي في جدول الأقسام وهو رقم القسم ، ثم نقوم بإنشاء حقل أجنبي في جدول الطلاب يعود لهذا الرئيسي ، وبالمثال يتضح المقال !

- إنشاء جدول الأقسام ، ويسمى هذا الجدول بالرئيسي (Mister) أو الأب ..

```
SQL> create table section
  2  ( sec_no number(2)
  3  primary key ,
  4  sec_name varchar2(20) not null
  5  );
```

نلاحظ أننا أنشأنا جدول الأقسام ؛ وفيه رقم القسم (sec_no) كمفتاح رئيسي ،واسم القسم (sec_name) بقيد ألا يكون فارغاً ..

- إنشاء جدول الطلاب ، ويسمى هذا الجدول بالتفصيل (Detail) أو الابن ..

```
SQL> create table student (
  2  std_no number(7)
  3  primary key,
  4  std_name varchar2 (30) not null,
  5  sec_no number(2) references
  6  section (sec_no)
  7  );
```

أنشأنا جدول الطلاب ؛ وفيه رقم الطالب كمفتاح رئيسي ..واسم الطالب بقيد ألا يكون فارغاً ..

لكن الأهم :

- أننا عرفنا حقل واسمه رقم القسم (sec_no الذي باللون الأخضر) وهذا أمر ضروري جداً أن نعرف الأجنبي أولاً في الجدول التفصيل ، ثم نبين أنه يؤشر إلى مفتاح رئيسي في جدول آخر بحجم هو نفس حجم رقم القسم في جدول الأقسام .. ،لأننا نعلم أن المفتاح الأجنبي يجب أن يكون تعريفه من نفس نوع بيانات وحجم المفتاح الرئيسي الذي يؤشر إليه ..ولكن ليس شرطاً أن يكون بنفس الإسم ..
- كتبنا كلمة references والتي تعني أنه يؤشر إلى المفتاح الرئيسي في جدول الأقسام Section وهو المفتاح (sec_no باللون الرمادي) ولا ننسى أن ماسبق كله هو بطريقة In line .

- مثال بطريقة Out line :

نقوم بإنشاء الجدول الثاني وهو التفصيل بهذه الطريقة ، لأن الجدول الأول لايهمنا إنشائه بأي طريقة ..

لا يمكننا حذف جدول الأقسام ، إلا بعد أن نحذف جدول الطلاب لوجود علاقة بينهما ، لنجرب ونكتب ..

```
SQL> drop table section;
```

ستظهر لنا رسالة خطأ ، مفادها أنه لا يمكننا حذف الجدول ،لارتباطه بحقل رئيسي بجدول الطلاب ..

فنقوم أولاً بحذف جدول الطلاب ، لإنشائه مرةً أخرى بطريقة Out line ..

```
SQL> drop table student;

Table dropped.
```

ولا نحذف جدول الأقسام بل نبقيه كما هو ..

وحتى نعرف الجداول التي لدينا الآن نكتب :

```
SQL>  select *from tab;
```

فتظهر لنا النتيجة التالية ..

```
TNAME                          TABTYPE CLUSTERID
------------------------------ ------- ---------
SECTION                         TABLE
```

الآن ننشئ جدول الطلاب كما أنشأناه بطريقة In line أما المفتاح الأجنبي فسننشئه إن شاء الله بطريقة Out line ..

```
SQL> create table student (
  2  std_no number(7)primary key,
  3  std_name varchar2 (30) not null,
  4  sec_no number(2),
  5  std_age number(2),
  6  constraint  fk_std_01
  7  foreign key (sec_no)
  8  references section (sec_no)
  9  );
```

نلاحظ أنه بالإضافة إلى وجود تعريف للمفتاح الرئيسي في هذا الجدول وهو رقم الطالب (std_no) ، احتوى هذا الجدول أيضاً مفتاحاً ثانوياً وهو رقم القسم (sec_no) ، فيسمح تعريف المفتاح الثانوي بتحديد الحقل المرجع في الجدول المرتبط به ، والذي يمكن أن يأخذ اسماً آخر ..

ولكي يكون تعريف المفتاح الثانوي سليماً ، يجب أن يكون كلا الحقلين من نفس نوع البيانات ونفس الحجم كما ذكرنا سابقاً..(جرب تعريف الحقل الثانوي بنوع آخر أو بحجم يختلف عن الحقل المؤشر إليه) .

وعند تعريف المفتاح الثانوي فإننا نخبر أوراكل بأننا نريد إنشاء تكامل مرجعي بين الحقل رقم القسم (sec_no) في الجدول student والجدول section .

وهذا الأمر يمنع الحقل في جدول الإبن : student ، من احتواء قيمة غير موجودة في العمود المرتبط به في الجدول الأب : section ..

ملاحظة هامة جداً :

عندما يكون لدينا جدولين أحدهما رئيسي (Mister) أي أب ، والذي تعتمد عليه الجداول الأخرى في البيانات ..والثاني تفصيلي (Detail) أي ابن ..

وكان هناك علاقة بين الجدولين بواسطة مفتاح أجنبي فإنه لا نستطيع أن نحذف البيانات الموجودة في الجدول الأب الـ (Mister) ..

وتوجد لهذه المشكلة طريقتان لحلها :

- الطريقة الأولى الأسهل والأضعف : حذف البيانات في الجدول الأبن الـ (Detail) ثم حذف البيانات في الجدول الأب الـ (Mister) ..

```
SQL> delete from student ;

SQL> delete from section ;
```

- الطريقة الثانية الأقوى والأمثل :وهي أنه عند تصميم الجدول الإبن الـ (Detail) فإننا نضيف بعد تعريف المفتاح الأجنبي العبارة التالية ..

```
On delete cascade
```

إذا نكتب الجدول الإبن الـ (Detail) وهو جدول الطلاب لدينا كما يلي ..

```
SQL> create table student (
  2  std_no number(7)primary key,
  3  std_name varchar2 (30) not null,
  4  sec_no number(2),
  5  std_age number(2),
  6  constraint  fk_std_01
  7  foreign key (sec_no)
  8  references section (sec_no)
     on delete cascade
  9  );
```

وعند كتابتنا لهذه العبارة بعد تعريف الحقل الثانوي ، فإن أوراكل لن يسمح فقط بحذف السجل المسار إليه من الجدول الأب وهو الأقسام (section) ، وإنما سيتتالى الحذف في الجدول الإبن أيضاً وهو جدول الطلاب (student) .

وأخيراً ..رجاء حار إليكم ..

تطبيق كل مثال أورده هنا مع كل قيد وبكلا الطريقتين ،و بدون نسخ الجدول ولصقه ، قم بكتابة كل تعليمة بنفسك حتى تستفيد وتتعلم ..

ملخص الدرس :

ملخص هذا الدرس هو زبدة الدرس ، وبالتفصيل الممل فاحرص على فهمه جيداً .

* القيـــدُ (Primary Key) أي المفتاح الرئيسي :

- وظيفته إعطاء حقل معين من عدة حقول في جدولٍ ماصفة المفتاح الرئيسي في هذا الجدول ، ويكون المفتاح الرئيسي في جدول ما ، وأجنبياً في الجداول الأخرى ..
- في بعض الحالات نظطر إلى إنشاء أكثر من حقل رئيسي في جدولٍ واحد ، ونستطيع عمل ذلك بالطريقة الغير خطية (Out line) فقط .

* القيـــدُ (Foreign Key) :

- وظيفته إعطاء حقل معين من عدة حقول في جدولٍ ما ؛ صفة المفتاح الأجنبي في هذا الجدول ، وهو عبارة عن حقل (صفة) أو أكثر تُضاف لجدول لربطه مع جدول آخر ، مع الالتزام بوجود مفتاح رئيسي مقابل .
- يجب أن المفتاح الأجنبي يجب أن يكون من نفس نوع بيانات (Data Type) المفتاح الرئيسي ..
- الصيغة العامة لتعريف حقل أجنبي في جدول بالطريقة الغير خطية هي :

constraint اسم القيد
اسم الحقل كما عرفناه في الجدول الحالي وهو جدول الأبن foreign key
(الحقل المرجع إليه وهو الرئيسي في الأب) الجدول الأب references
on delete cascade

وهذا شرح الصيغة بالتفصيل الممل ...

- constraint كلمة تعني أن ما بعدها سيكون اسماً لقيدٍ معين على حقل معين ، وهي محجوزة كما عرفناها سابقاً وتكتب في الطريقة الغيرخطية فقط ..
- foreign key كلمة تعني أن ما بعدها سيكون اسماً لقيدِ المفتاح الثانوي (الأجنبي) .
- اسم القيد وهنا نكتب اسم القيد ، وكلما كان اسم القيد معبراً وسهلاً كان أفضل ، فمثلاً أعلاه في مثال الطريقة الثانية ، كان اسم القيد هو fk_std_01 فالجزء الأول fk

يدل على أن القيد هو قيد المفتاح الثانوي ، أما std فهي تعني اسم الجدول ومكان القيد الذي وضعناه وهنا جدول الطلاب مكان هذا المفتاح ، وأخيراً فـ 01 فتعني رقم القيد ، أي أنه هنا القيد الأول من نوع المفتاح الثانوي في جدول الطلاب ..فإذا كان لدينا أكثر من مفتاح ثانوي فسيكون الترتيب يبدأ من الرقم 02 أي بعد الأول مباشرة ..ولكن لماذا هذا الحرص الشديد على اسم القيد ، ولماذا نكتبه بهذه الطريقة ؟!

الجواب منطقي ... أنه عندما تنشىء مستقبلاً بإذن الله عشرات وقد تصل إلى مئات الجداول ، وتريد أن تغير في قيد حقل معين أو أن تلغي قيد حقل معين ..فإنك ستمكث وقتاً ليس باليسير من أجل ذلك ..أما عند تسميته بهذه الطريقة فستصل إليه بسرعة تامة إن شاء الله وتعدله في أسرع وقت ..

- اسم الحقل كما عرفناه في الجدول الحالي وهو جدول الأبن ، حيث نقوم بكتابة اسم هذا المفتاح كما عرفناه في الجدول الحالي الذي نعرف فيه هذا المفتاح الأجنبي وهو جدول الابن ..ويجب أن يكون تعريف هذا الحقل من نفس نوع بيانات وحجم المفتاح الرئيسي الذي يؤشر إليه في الجدول الأب .

أما References تعني أن هذا الحقل يؤشر إلى المفتاح الرئيسي في الجدول الأب ..

وهو يخبر أوراكل (إن جاز التعبير) إنشاء تكامل مرجعي بين الحقل المفتاح الأجنبي في الجدول الابن ..والحقل المفتاح الرئيسي في الجدول الأب (راجع المثال الأخير) ..

- الجدول الأب أي أن المفتاح الأجنبي يؤشر للمفتاح الرئيسي في الجدول الأب المسمى (..)..
- الحقل المرجع إليه وهو الرئيسي في الأب ، أي المفتاح (الحقل) الرئيسي في الجدول الأب الذي يؤشر إليه الأجنبي الذي أنشناه ..
- وأخيراً عبارة on delete cascade فائدتها أنه عندما نريد حذف البيانات من الجدول الاب ، لن نستطيع الحذف لوجود العلاقة بين الجدولين الأب والابن ، فعند كتابتنا لها بعد تعريف الحقل الثانوي ، نستطيع حف البيانات ، ولن يسمح أوراكل فقط بحذف السجل المسار إليه من الجدول الأب ، وإنما سيتتالى الحذف في الجدول الإبن أيضاً .

****

وسامحونا على التقصير ، وفق الله الجميع لما يحبه ويرضاه ..

واستودعكم الله الذي لاترد ودائعه ..

وإلى لقاء قريب بإذن الله تعالى ..

الأسيـــــــــــــــــــــــف

Email:amaar1422@hotmail.com

بسم الله الرحمن الرحيم

الدرسُ السابع | إضافة الحقول وتعديلها في الجداول

بعد أن أخذنا أمر إنشاء الجداول (CREATE TABLE) وعرفنا كيفية إنشاء الجداول بحقولها وبقيودها المختلفة ، ثم تعلمنا كيفية حذفها (DROP TABLE) ، تبقى لنا أخيراً كيفية الإضافة إليها والتعديل فيها (ALTER TABLE) ، وهذا التعديل هو في صميم بنية الجدول ، وليس في بيانات الجدول ، لأن التعديل في البيانات يتم عن طريق اللغة الفرعية الثانية من SQL وهي DML .

- **أمر تعديل الجداول** ALTER TABLE :

بواسطة هذا الأمر نستطيع إنجاز الكثير من العمليات وهي :

- إضافة حقل جديد إلى جدول منشأ سابقاً .
- التعديل في صفة حقلٍ ما .
- الإضافة و التعديل في قيود منشأة على عدة حقول مسبقاً .
- تأهيل القيود وإعادة تأهيليها.
- حذف القيود من الجداول ..

وسنتطرق في درسنا هذا للعمليتين الأولى ..وأما الثلاث الأخرى ففي الدرس القادم بإذن الله ..

- ***إضافة حقل جديد إلى جدول منشأ سابقاً :***

عندما ننشئ الجداول ونقوم بإدخال البيانات إليها ، نحتاج في بعض الأحيان لإضافة حقل أو عدة حقول لهذا الجداول ، ولأجل ذلك نستخدم الصيغة التالية :

```
ALTER TABLE اسم الجدول ADD
        اسم الحقل   نوع البيانات  (طولها)  [ فارغ أو غير فارغ أو قيود ]
);
```

وهذه الصيغة تستخدم لإضافة حقل أو حقول جديدة بناءً على الكلمة **ADD** .

وحتى نفهم هذه الصيغة جيداً ، سنقوم بإضافة حقل جنس الطلاب الذين يدرسون في هذه المدرسة S_SEX (ذكر أم أنثى) لأول جدول أنشأناه سوياً وهو SCHOOL ..وسنقوم أولاً باستعراض خصائص الجدول SCHOOL من خلال الأمر التالي ..

```
SQL> desc school;
```

إذا لم يكن الجدول موجوداً لديك ستظهر لك العبارة التالية ..

```
Object does not exist.
```

وقم بكتابة هذه الأوامر لإنشاء الجدول كما تعلمنا سابقاً ..

```
CREATE  TABLE  SCHOOL
(
   S_NO      NUMBER  (10) ,
   S_NAME  VARCHAR2 (30) ,
   S_TYPE    CHAR  (15) ,
   S_LOC     VARCHAR2  (30) ,
   S_DATE    DATE
) ;
```

وبعد أن قمت بإنشاء هذا الجدول استعرض خصائصه ..

```
SQL> DESC SCHOOL;
```

وستظهر لنا مواصفات جدول المدرسة كمايلي ..

```
Name                      Null?    Type
------------------------------ -------- ----
S_NO                           NUMBER(10)
S_NAME                         CHAR2(30)
S_TYPE                         CHAR(15)
S_LOC                          CHAR2(30)
S_DATE                         DATE
```

والآن نريد إضافة حقل جنس الطلاب الذين يدرسون في هذه المدرسة S_SEX (ذكر أم أنثى) لهذا الجدول فنكتب الأمر التالي ..

```
alter table school add
 ( s_sex varchar2(10)
) ;
```

فتظهر لنا العبارة التالية ..والدالة على أن الحقل الجديد تمت إضافته للجدول ..

Table altered.

وحتى نتأكد من هذه الخطوة نستعرض خصائص الجدول كما تعلمنا سابقاً ..

SQL> DESC SCHOOL;

وستظهر لنا مواصفات جدول المدرسة كمايلي ..

Name	Null?	Type
S_NO		NUMBER(10)
S_NAME		VARCHAR2(30)
S_TYPE		CHAR(15)
S_LOC		VARCHAR2(30)
S_DATE		DATE
S_SEX		VARCHAR2(10)

ولاحظ الحقل الجديد وهو S_SEX قد تمت إضافته للجدول ..

- ***التعديل في مواصفات حقل ما في جدول منشأ مسبقاً أيضاً :***

نحتاج أحياناً لتعديل مواصفات حقل ما ، مثل زيادة سعته التخزينية أو تغيير صفته ، وهذا ممكن لكن بشروط نفهمها من خلال القاعدة العامة :

********** (الزيادات مقبولة دوماً أما الإنقاص فيؤدي إلى مشاكل)**********

وكأمثلة على الزيادات المقبولة ..

- زيادة حجم حقل من النوع VARCHAR2 أو النوع CHAR .
- زيادة حجم حقل من النوع NUMBER .
- زيادة عدد الحقول في جدول .

وكأمثلة على عمليات الإنقاص المقبولة ..

- إنقاص حجم حقل NUMBER .
- إنقاص حجم عمود VARCHAR2 أو CHAR .

لكن شرط الإنقاص الهام هو : أنه عند إنقاص حجم عمود يجب أن يكون هذا الحقل فارغاً من البيانات .

وصيغة تعديل مواصفات حقلٍ ما في جدول هي كالتالي ..

```
ALTER TABLE اسم الجدول MODIFY
(
      الخاصية أو الخواص    اسم الحقل
);
```

والآن نريد مثلاً تعديل خاصية الحقل الجديد S_SEX الذي أنشأناه مؤخراً من النوع VARCHAR2 إلى النوع CHAR .. فنقوم بكتابة الأمر التالي :

```
Alter table school modify
( s_sex char(10)
);
```

فتظهر لنا العبارة التالية ، والتي تفيد أنه تم التعديل ..

```
Table altered.
```

ومن الممكن أن نستعرض خصائص الجدول لنلاحظ تغير صفة الحقل SEX من النوع VARCHAR2 إلى النوع CHAR ..

ملحوظة هامة جداً : أنه إذا كان الجدول به بيانات وتم تغيير نوع بيانات الحقل من النوع الحرفي إلى النوع الرقمي أو العكس ، فإن اللغة تعطي (رسالة خطأ) تعني أن الحقل يجب أن يكون خالياً من البيانات أو في حالة أن البيانات المخزنة في الجدول لا تتناسب مع مواصفات الحقل بعد التعديل

ملخص الدرس :

- أمر تعديل الجداول ALTER TABLE :

وبواسطة هذا الأمر نستطيع إنجاز الكثير من العمليات وهي :

- إضافة حقل جديد إلى جدول منشأ سابقاً .
- التعديل في صفة حقلٍ ما .
- الإضافة و التعديل في قيود منشأة على عدة حقول مسبقاً .
- تأهيل القيود وإعادة تأهيليها.
- حذف القيود من الجداول ..

- صيغة إضافة حقل جديد إلى جدول منشأ سابقاً هي :

```
ALTER TABLE اسم الجدول ADD
      اسم الحقل   نوع البيانات   (طولها)  [ فارغ أو غير فارغ أو قيود ]
);
```

وهذه الصيغة تستخدم لإضافة حقل أو حقول جديدة بناءً على الكلمة **ADD** .

- صيغة التعديل في مواصفات حقل ما في جدول منشأ مسبقاً أيضاً هي :

```
ALTER TABLE اسم الجدول MODIFY
(
      اسم الحقل    الخاصية أو الخواص
);
```

وقاعدة التعديل في مواصفات حق هي :

...(الزيادات مقبولة دوماً أما الإنقاص فيؤدي إلى مشاكل)...

****

وفي الدرس القادم بإذن الله سنستكمل العمليات الثلاث الأخرى للأمر ALTER .. فإلى لقاء قريب بإذن الله تعالى .وسامحونا على التقصير ، وفق الله الجميع لما يحبه ويرضاه ..

محبكم في ربكم

الأسيـــــــــــــــــــــــــــــــف

Email:amaar1422@hotmail.com